SERVICE POLICY FOR READ IN MIRRORED CACHE

Cross Reference to Related Applications

This application is a continuation in part of U.S. Patent application No. 09/676,686 filed on September 29, 2000 (pending).

Background of The Invention

1. Field of the Invention

This application relates to the field of computer data storage and more particularly to the field of using a cache memory in a computer data storage device.

2. Description of Related Art

Host processor systems may store and retrieve data using a storage device containing a plurality of host interface units, disk drives, and disk interface units. Such storage devices are provided, for example, by EMC Corporation of Hopkington, Mass. and disclosed in U.S. Patent No. 5,206,939 to Yanai et al., U.S. Patent No. 5,778,394 to Galtzur et al., U.S. Patent No. 5,845,147 to Vishlizzky et al., and U.S. Patent No. 5,857,208 to Ofek. The host systems access the storage device through a plurality of channels provided therewith. Host systems provide data and access control information via the channels of the storage device and the storage device provides data to the host systems also through the channels. The host systems do not address the disk drives of the storage device directly, but rather, access what appears to the host systems as a plurality of logical disk units. The logical disk units may or may not correspond to the actual disk drives.

Performance of such a storage system may be improved by using a cache. In the case of a disk drive system, the cache may be implemented using a block of semiconductor memory that has a relatively lower data access time than the disk drive. Data that is accessed is advantageously moved from the disk drives to the cache so that the second and subsequent accesses to the data may be made to the cache rather than to the disk drives. Data that has not been accessed recently may be removed from the cache to make room for new data. Often such cache accesses are transparent to the host systems requesting the data.

In instances where the host systems write data to the disk, it may be efficient to have the write operation initially occur only in the cache. The data may then be transferred from the cache back to the disk at a later time, possibly after subsequent read and write operations. Transferring the modified cache data to the disk is referred to as "destaging".

If the cache memory fails after one or more write operations but prior to destaging the modified cache data to the disk, then the disk data may not match the data that was written by the host system. Such a situation may be especially troublesome in instances where the use of the cache is transparent to the host, i.e., in systems where the host system writes data and the write operation is acknowledged by the storage device (because the data is successfully written to the cache), but then the data is never appropriately transferred to the disk because of cache failure. Numerous solutions have been proposed to handle cache failures.

U.S. Patent No. 5,437,022, U.S. Patent No. 5,640,530, and U.S. Patent No. 5,771,367, all to Beardsley et al, disclose a system having two, somewhat – independent, "clusters" that handle data storage. The clusters are disclosed as being designed to store the same data. Each of the

clusters includes its disks own cache and non-volatile storage area. The cache from one of the clusters is backed up to the non-volatile data storage area of the other cluster and vice versa. In the event of a cache failure, the data stored in the corresponding non-volatile storage area (from the other cluster) is destaged to the appropriate disk. However, this system requires, in effect, a duplicate backup memory for each of the caches and also provides that whenever data is written to one of the caches, the same data needs to be written to the corresponding non-volatile storage in the other cluster. In addition, since each cluster includes a cache and a non-volatile storage, thus having two redundant clusters requires four memories (one cache for each of the clusters and one non-volatile storage for each of the clusters).

It is desirable to have a system that provides sufficient redundancy in the case of failure of a cache element without unduly increasing the complexity of the system or the number of elements that are needed.

Summary Of The Invention

In accordance with a first aspect of the invention is a method of managing data in a cache. A first cache memory is provided that contains data. A second cache memory is provided that contains data wherein at least some of the data in the first cache memory is the same as at least some of the data in the second cache memory. In response to a request for data that is stored in both the first cache memory and the second cache memory, one of the cache memories is chosen to use to obtain the data according to an access balancing technique.

In accordance with another aspect of the invention is a computer program product for managing data in a cache. Machine executable code is included for: providing a first cache memory that contains data; providing a second cache memory that contains data wherein at least some of the data in the first cache memory is the same as at least some of the data in the second cache memory; and, in response to a request for data that is stored in both the first cache memory and the second cache memory, choosing one of the cache memories to use to obtain the data according to an access balancing technique.

In accordance with yet another aspect of the invention is a system for managing data in a cache. A first cache memory includes data. A second cache memory includes data wherein at least some of the data included in the first cache memory is the same as at least some of the data of the second cache memory. Cache selection hardware is included for selecting, in response to a request for data that is stored in both the first cache memory and the second cache memory, which one of the first and second cache memories to use to obtain the data in accordance with an access balancing technique.

Brief Description Of Drawings

Fig. 1A shows a pair of cache memories where each is coupled to a pair of buses in an embodiment of the system described herein.

Fig. 1B shows a pair of cache memories coupled to a single bus in an other embodiment of the system described herein.

Fig. 2 is a schematic diagram illustrating a host system coupled to a storage system containing a pair of cache memories and a disk storage area according to the system described herein.

Fig. 3 is a table that may be used to determine primary and secondary cache memories for each of the slots of the disk storage area of the system described herein.

Fig. 4 shows a pair of cache memories having slots and control data associated therewith according to the system described herein.

Fig. 5 is a flow chart illustrating steps performed in connection with failure of the hardware associated with one of the pair of cache memories.

Fig. 6 is a flow chart illustrating steps performed in connection with a host accessing data in the cache memories.

Fig. 7A is a flow chart illustrating steps performed in connection with providing data from the disk storage area to the cache memories according to the system described herein.

Fig. 7B is a flow chart illustrating steps performed in connection with handling data that is modified after the data has been read into the cache according to the system described herein.

Fig. 8 is a flow chart illustrating steps performed in connection with recovery after failure and replacement of the hardware associated with one of the cache memories.

Fig. 9 is a partial flow chart illustrating a portion of the steps performed in connection with a host accessing data in the cache memories.

Fig. 10 is a partial flow chart illustrating a portion of the steps performed in connection with a host accessing data in the cache memories.

Fig. 11 is a partial flow chart illustrating a portion of the steps performed in connection with a host accessing data in the cache memories.

Fig. 12 is a partial flow chart illustrating a portion of the steps performed in connection with a host accessing data in the cache memories.

Fig.13 is a partial flow chart illustrating a portion of the steps performed in connection with a host accessing data in the cache memories.

Fig. 14 illustrates specialized hardware for providing the functionality illustrated in connection with Fig.'s 9-13.

Detailed Description of the Preferred Embodiment(s)

Referring to Fig. 1A, a schematic diagram 20 shows a first cache memory 22, and a second cache memory 24 each coupled to a first bus 26 and a second bus 28. The cache memories 22, 24 and the buses 26, 28 may be part of a larger system, such as a data storage device provided by EMC Corporation of Hopkinton, Mass. Data may be written to and read from the memories 22, 24 via the busses 26, 28. The first memory 22 may be coupled to the first

bus 26 via a first controller 32 and may be coupled to the second bus 28 via a second controller 34. Similarly, the second memory 24 may be coupled to the first bus 26 via a third controller 36 and may be coupled to the second bus via a fourth controller 38. The busses 26, 28 may be deemed "odd" and "even" for reference purposes. Similarly, the memories 22, 24 may be deemed "top" and "bottom".

In some embodiments, the buses 26, 28 are entirely redundant and each of the buses 26, 28 is coupled to all of the disk controllers (not shown) and host interface units (not shown) of the corresponding storage device. In other embodiments, each of the buses 26, 28 may be connected to a different set of host interface units and disk controllers, possibly with some overlap. Alternatively still, it is possible to have one of the buses 26, 28 couple to all of the host interface units while the other one of the buses 26, 28 is coupled to all of the disk controllers. Configuring and managing the redundancy of the buses 26, 28 may be provided according to a variety of functional factors known to one of ordinary skill in the art, and the system described herein is adaptable to any such configuration. Note that it is possible to further subdivide the busses 26, 28 and the components connected thereto to reduce the likelihood of bringing the whole system down in connection with failure of a bus or of a component thereof.

Referring to Fig. 1B, a schematic diagram 30 shows an alternative embodiment where the first cache memory 22 and the second cache memory 24 are both coupled to a single bus 26'. In the embodiment of Fig. 1B, the bus 26' may be coupled to all of the host interface units and all of the disk controllers of the corresponding storage device. The system described herein may be configured with either the embodiment of Fig. 1A, the embodiment of Fig. 1B, or other configurations of one or more buses coupled to the cache memories 22, 24.

Referring to Fig. 2, a schematic diagram 40 illustrates a storage system 41 and the flow of data between the cache memories 22, 24, a disk storage area 42, and a host system 44. Data flows between the first cache memory 22 and the disk storage area 42 and flows between the first cache memory 22 and the host system 44. Similarly, data flows between the second cache memory 24 and the disk storage area 42 and between the second cache memory 24 and the host system 44. Specific control of the data between the hosts system 44, the cache memories 22, 24, and the disk storage area is described elsewhere herein.

Referring to Fig. 3, a table 52, which is part of the data that is used to control operation of the storage device 41, indicates portions T1, T2. . . TN of the cache memories 22, 24 that are to be designated as primary storage areas. In one embodiment, the cache memories 22, 24 are mapped alternatively so that, for example, a first set of portions may be designated as primary for the cache memory 22 while a second set of portions may be designated as primary for the cache memory 24, where the first and second sets are interposed. In some embodiments, the portions are 1/4 Gigabyte in size, although it will be apparent to one of ordinary skill in the art that the invention may be practiced using other sizes. The purpose of the mapping is discussed in more detail elsewhere herein.

Referring to Fig. 4, a schematic diagram illustrates the cache memories 22, 24 in more detail. In some embodiments, each of the cache memories 22, 24 is implemented using separate hardware. Each of the memories 22, 24 is shown as containing a plurality of slots S1, S2,. . .SZ which, for embodiments discussed herein, provide storage for a sector of the disk storage area 42. For the embodiments illustrated herein, one sector equals eight blocks and one block equals five hundred and twelve bytes. However, it will be apparent to one of ordinary skill in the art

that other sizes may be used without departing from the spirit and scope of the system described herein.

Associated with each of the slots may be specific control data elements C1, C2,. . .CZ, so that control data element C1 is associated with slot S1, control data element C2 is associated with slot S2, and so forth. For the system described herein, there is control data associated with each block and each sector. In addition, in some embodiments, it is possible to indicate that particular blocks of data are write pending, rather than indicating that an entire sector, to which the block belongs, is write pending. However, the discussion herein will emphasize control data and the write pending state for sectors.

Each of the slots represents data that is read from the disk storage area 42 and stored in one or both of the cache memories 22, 24. The control data for each of the slots indicates the state of the data in the slot. Thus, for example, the control data element for a slot can indicate that the data has been read from the disk storage area 42 but not written to by the host 44 (i.e., not modified by the host 44). Alternatively, the control data element for a slot could indicate that the data in the slot has been written to by the host 44 since being read from the disk storage area 42 (i.e., write pending). Note that, generally, data that is read from the disk storage area 42 but not subsequently modified may be eliminated from the cache without any ultimate loss of data since the data in the memories 22, 24 is the same as the data in the disk storage area 42. On the other hand, data that is write pending (i.e., modified while in the memories 22, 24 after being read from the disk storage area 42) is written back to the disk storage area 42 for proper data synchronization. Note also that the control data could indicate that the associated slot contains data that is the same in both of the memories 22, 24, which could occur, for example, either

when the data is write pending or immediately after data that is write pending is written to the disk.

In one embodiment, data that is read from the disk storage area 42 is written to one or the other of the memories 22, 24. The shading of the slots in the memories 22, 24 in Fig. 4 indicates that a slot has been designated as a secondary slot. Thus, for example, the slots S1, S2 . . . SN of the cache memory 22 are designated as secondary slots while the slots SO, SP, . . . SQ of the cache memory 24 are designated as secondary slots. Conversely, the slots SO, SP, . . . SQ of the cache memory 22 are designated as primary slots while the slots S1, S2, . . . SN of the cache memory 24 are designated as primary slots.

In one embodiment, data that is read from the disk storage area 42 is written only to the corresponding primary slot and, at least initially, is not written to the secondary slot. Thus, for example, if a sector of data is to be provided in slot S1, the data is read from the disk and is initially written only to the cache memory 24. Similarly, data from the disk designated for slot SP is initially written only to the cache memory 22. The hardware may be used, in a conventional manner, to control writing to one of the cache memories 22, 24 or writing to both of the memories 22, 24 simultaneously (and/or with a single command). Similarly, the hardware may control which of the memories 22, 24 is read.

If an event occurs causing data in the cache memories 22, 24 to change (such as a write from the host 44), then the modified data is written to both the primary memory and to the secondary memory. For example, data that is designated for slot S1 is initially written from the disk storage area 42 only to the cache memory 24. However, if a subsequent operation occurs

that causes the data in slot S1 to change (i.e., a write by the host 44 to the portion of the disk storage area 42 corresponding to slot S1), then the data in slot S1 is modified according to the write operation which writes data to both of the memories 22, 24. Thus, data that is write pending exists in both of the cache memories 22, 24. Note that, in some instances, unmodified but related data in a slot may be copied from one of the memories 22, 24 to the other one of the memories 22, 24.

The state of the data in the slots is indicated by the control data. Thus, in the case of data that has not been modified, the corresponding control data element indicates that the data has not been modified while, in the case of data that has been modified, the corresponding control data element indicates that the data is write pending. The control data for the slots is written to both of the cache memories 22, 24. Thus, in the event of loss of the hardware associated with one of the cache memories 22, 24, the entirety of the control data will exist in the non-failing one of the cache memories 22, 24. Stated differently, the control data information in one of the cache memories 22, 24 is identical to the control data information in the other one of the cache memories 22, 24.

Note that any data that is write pending in the cache is provided in both of the cache memories 22, 24. On the other hand, data that does not need to be written back to the disks (i.e., data that has not been modified by the host 44) is stored in only one of the cache memories 22, 24. Storing the data in only one of the cache memories 22, 24 is an optimization that can increase performance by requiring only one write to one of the cache memories 22, 24 in certain instances, while providing a mechanism where write pending cache data is written to both of the cache memories 22, 24. In addition, note that, as discussed above, identical data may be stored

in corresponding slots in both of the memories 22, 24 even though the data is not write pending. This may occur, for example, immediately after write pending data is copied to the disk.

Referring to Fig. 5, a flow chart 60 illustrates steps performed in the event that the hardware associated with one of the cache memories 22, 24 fails. Implementing each of the cache memories 22, 24 with separate hardware increases the likelihood that failure of the hardware for one of the cache memories 22, 24 will not occur at the same time as failure of the hardware for an other one of the cache memories 22, 24. Detection of the failure of one of the cache memories 22, 24 is provided in a straightforward manner, such as described in U.S. Patent No. 5,742,501 to Dewey et al., which is incorporated by reference herein. Note that detection of a failure may occur during an initial self test.

Processing begins at a first step 62 where a pointer is set to point to the first slot of the good cache memory (i.e., the one of the cache memories 22, 24 that has not failed). Following the step 62 is a test step 64 where it is determined if the data stored in the slot that is pointed to is duplicated in the memories (i.e., is the same for both of the memories 22, 24). As discussed above, this is indicated by the corresponding control data for the slot. Note that this information is available irrespective of whether the slot of the non-failing one of the cache memories 22, 24 is a primary or a secondary storage area, since all of the control data is duplicated between the cache memories 22, 24, as discussed elsewhere herein.

If it is determined at the test step 64 that the data for the slot is not the same for both of the memories 22, 24, then control passes from the test step 64 to a test step 66 where it is determined if the non-failing cache memory (i.e., the one of the cache memories 22, 24 that is

being examined) is the primary storage area for the data. If it is determined at the test step 66 that the slot being examined is not the primary storage area for the data (and thus the data is not stored in the non-failing cache memory), then control passes from the test step 66 to a step 68 where the control data for the slot is modified to indicate that the corresponding data is not in the cache. The step 68 is executed because the data corresponding to the slot being examined is stored in the failed one of the cache memories 22, 24 and thus, effectively, is no longer in the cache.

Following the step 68 is a step 70 where the next slot of the non-failing cache is pointed to in order to be examined on the next iteration. Following the step 70 is a test step 72 where it is determined if processing is complete (i.e., no more slots remain to be examined). If it is determined at the test step 72 that there are more slots to examine, then control transfers back to the step 64 to process the next slot.

Note that the step 70 is also reached from the step 64 if it is determined that the data is the same in both of the memories 22, 24 and that the step 70 is also is reached from the test step 66 if it is determined that the data, although not the same in both of the memories 22, 24, is stored in the non-failing one of the cache memories 22, 24. This is because, in either of these cases, it is not necessary to mark the control data for the slot being examined as indicating that the data is not in cache at the step 68.

Referring to Fig. 6, a flow chart 80 illustrates steps performed in connection with a read operation executed by the host where the data being read is in one or both of the cache memories 22, 24. Note that, if the hardware for one of the cache memories 22, 24 fails, then only one of

the cache memories 22, 24 is used for all data read and write operations. However, in the course of normal operation, both of the cache memories 22, 24 are used to store data.

Processing begins at a first step 82 where the control data for the data being accessed is obtained. Note that, as discussed elsewhere herein, the control data is duplicated between the cache memories 22, 24. Thus, the selection of one of the cache memories 22, 24 from which to read the control data at the step 82 may be random, or may be alternative (i.e., round robin), or may be some other scheme that may or may not provide for balancing accesses and/or performance enhancement between the cache memories 22, 24. In some embodiments, it may be desirable to provide load balancing and/or performance enhancement in connection with read operations.

Following the step 82 is a step 84 where it is determined if the data is the same in both of the memories 22, 24. As discussed above, this information may be provided by the corresponding control data element. If it is determined at the test step 84 that the data is the same in both of the memories 22, 24, then the data may be read from either one of the cache memories 22, 24. Thus, if it is determined at the step 84 that the data is the same in both of the cache memories 22, 24, then control passes from the step 84 to a step 86, where the data is read from either of the cache memories 22, 24. In some embodiments, at the step 86 the data is read from the one of the cache memories 22, 24 that is used at the step 82 to obtain the control data. In other embodiments, at the step 86 the data is read from the one of the cache memories 22, 24 opposite to the one of the cache memories 22, 24 that is used at the step 82. Following the step 86, processing is complete.

If it is determined at the test step 84 that the data that is not the same in both of the cache memories 22, 24, then control passes from the test step 84 to a test step 88 where the data is read from the primary cache for the data. The distinction between primary and secondary cache storage is discussed elsewhere herein. Following the step 88, processing is complete.

Referring to Figure 7A, a flow chart 100 illustrates steps performed in connection with providing data from the disk storage area 42 to the cache memories 22, 24. At a first step 102, it is determined which of the memories 22, 24 is the primary storage area for the data. Following the step 102 is a step 104 where the data is copied from the disk storage area 42 to the one of the memories 22, 24 corresponding to the primary storage area. Following the step 104 is a step 106 where the corresponding control data element, for both of the cache memories 22, 24, is marked to indicate that the corresponding data is in cache, thus indicating that the data has been read in to the cache. As discussed above, the control data for each of the slots of the cache memories 22, 24 is duplicated. Thus, the control data element for any slot in one of the cache memories 22, 24 is made to equal the control data for the slot in the other one of the cache memories 22, 24 by writing the control data to both of the memories 22, 24 at the step 106. Following the step 106, processing is complete.

Referring to Fig. 7B, a flow chart 110 indicates steps performed in connection with the data in the cache that has been modified (e.g., by a write from the host 44). Note that the steps of the flow chart 110 may be executed some time after the data has been read from the disk storage area 41 in to the cache or may never be executed at all for some of the cache data.

At a first step 112, the block of data that is being modified (i.e., by the host 44) is written to both of the cache memories 22, 24. In each instance where data is modified, it is written to both of the caches 22, 24. However, the first time data from a slot (sector) is modified while in cache, other steps are also taken, as described below.

Following the step 112 is a step 114 where the remainder of the sector that includes the modified block is copied from the primary cache to the secondary cache. As discussed above, the embodiments disclosed herein operate a sector at a time, although is would be apparent to one of ordinary skill in the art how to adapt the system to operate using different size data increments, such as a block. Thus, if the control data is provided on a per block basis, and if the cache holds and manipulates data in units of blocks, then it may be possible to forego the step 114. Note also that if the control data indicates that the data for the sector is the same in both of the memories 22, 24, then the step 114 may be omitted, since there would be no need to copy data that is already the same.

Following the step 114 is a step 116 where the control data for the particular slot, in both of the memories 22, 24, is marked to indicate that the slot is write pending, indicating that the data has been modified while stored in the cache. As discussed above, the control data is written to both the primary and secondary storage areas. Following step 116, processing is complete. Note that when the write pending data is destaged, the control data may indicate that the data is no longer write pending although the control data may also indicate that the sector data in both of the memories 22, 24 is identical.

The cache memories 22, 24 may be configured as separate memory boards (separate hardware) and, in some embodiments, may each have their own power supply. Using separate hardware for each of the cache memories 22, 24 decreases the likelihood that both of the cache memories 22, 24 will fail simultaneously. Thus, when the hardware for one of the cache memories 22, 24 fails, the process set forth in Fig. 5, discussed above, may be executed to reconfigure the system to operate using a single cache memory.

Following a failure, it may be possible to replace the failed hardware while the system is operational using techniques for doing so that are discussed, for example, in U.S. patent No. 6,078,503 to Gallagher et al., which is incorporated by reference herein. However, once the hardware for the failed memory board is replaced, it is necessary to have a plan for recovery so that the system can use both of the cache memories 22, 24 in connection with normal operation.

Referring to Fig. 8, a flow chart 120 illustrates steps performed after the hardware for one of the cache memories has failed. Processing begins at a first step 122 which determines if the failed memory hardware has been replaced. The test step 122 represents waiting until new, operational, hardware for the failed memory board is installed. Thus, until the hardware for the failed memory is replaced, the step 122 loops back on itself. Stated differently, the remaining steps of the flowchart 120 are not performed unless and until the failed memory board is successfully replaced.

Once the hardware for the failed memory has been replaced, control passes from the step 122 to a step 124 where the system is configured to write all data to both of the cache memories

22, 24. That is, every time data is read from the disk storage area 42 to the cache, or data that is in the cache is modified by the host 44, the data is written to both of the cache memories 22, 24.

Following the step 124 is a step 126 where background copying is begun. Background copying refers to copying data from the non-failing one of the cache memories 22, 24 to the other one of the cache memories 22, 24 that corresponds to the new memory hardware. Background copying occurs when the cache is otherwise not being used. Thus, the steps 124, 126 cause the cache memories 22, 24 to eventually become duplicates of each other.

Following the step 126 is a test step 128 which determines if background copying is complete. If not, the step 128 loops back on itself to wait for completion. Otherwise, once background copying is complete, the cache memories 22, 24 are duplicates of each other and control passes from the step 128 to a step 130, where the system is reconfigured to operate in the usual manner as discussed above in connection with the Fig. 6, Fig. 7A, and Fig. 7B. Thus, when the hardware for one of the cache memories 22, 24 fails, the system operates with the single, non-failing cache memory. However, once the recovery process set forth in Fig. 8 is completed, then the system is reconfigured to have a primary and secondary cache and to operate in the usual manner, as discussed above.

When the same data is stored on both of the cache memories 22, 24, it is possible for the host system 44 to access the data from either of the memories 22, 24. Accordingly, in some instances, it may be possible to enhance performance by balancing access between the cache memories 22, 24. Depending upon the hardware configuration, it may be possible to access one of the cache memories 22, 24 while simultaneously accessing the other one of the cache

memories 22, 24. Thus, balancing the accesses could enhance performance by increasing the number of simultaneous accesses and correspondingly decreasing the number of (inherently inefficient) serial accesses to the same one of the cache memories 22, 24.

Referring to Fig.'s 9-13, a plurality of partial flow charts 80a - 80e illustrate modifications to the read operation illustrated in Fig. 6 and described above. The modifications facilitate balancing accesses of the data to increase throughput and enhance performance.

The partial flow chart 80a of Fig. 9 illustrates an alternative step 82' that corresponds to the step 82 of Fig. 6 where the control data is accessed. As discussed above, for embodiments illustrated herein, the control data is the same for both of the cache memories 22, 24. The step 82' represents choosing which of the cache memories 22, 24 to use for accessing the control data based on a statistical analysis. Although many different types of statistical analysis techniques could be used and would be apparent to one of ordinary skill in the art, for one embodiment of the invention, the statistical analysis of Fig. 82' simply tallies the number of accesses of each of the memories 22, 24 over a predetermined amount of time, such as one second. That is, if for the one second prior to the current access, the number of accesses of the cache memory 22 is N and the number of accesses to the cache memory 24 is M, then, if $N < M$, the cache memory 22 will be used, since the cache memory 22 is the one of the cache memories 22, 24 having the least number of previous accesses in the previous one second.

Referring to Fig. 10, the partial flow chart 80b illustrates another alternative step 82'' that corresponds to the step 82 of Fig. 6 where the control data is accessed. The step 82'' represents accessing the control data using a conventional round robin technique (discussed above), where

one of the cache memories 22, 24 is chosen based on which of the cache memories 22, 24 was previously not chosen. That is, if in the most recent iteration the cache memory 22 was used, then in the current iteration the cache memory 24 is used. Similarly, if in the most recent iteration the cache memory 24 was used, then in the current iteration the cache memory 22 is used.

Referring to Fig. 11, the partial flow chart 80c illustrates an alternative step 86' that corresponds to the step 86 of Fig. 6 where disk data that is the same in both of the cache memories 22, 24 is accessed. For the step 86', the disk data is read from the one of the cache memories 22, 24 that is opposite to the one of the cache memories 22, 24 used to access the control data. That is, irrespective of the technique for selecting which of the cache memories 22, 24 to use to access the control data, the step 86' represents reading the disk data from the other one of the cache memories 22, 24. Thus, for example, if the control data is read from the cache memory 22, then the step 86' represents reading the disk data from the cache memory 24. Similarly, if the control data is read from the cache memory 24, then the step 22 represents reading the disk data from the cache memory 22.

Referring to Fig. 12, the partial flow chart 80d illustrates an alternative step 86'' that corresponds to the step 86 of Fig. 6 where disk data that is the same in both of the cache memories 22, 24 is accessed. The step 86'' represents selecting the one of the cache memories 22, 24 from which to read the disk data using a simple statistical technique similar to that discussed above in connection with the step 82' of Fig. 9 used to select which of the cache memories 22, 24 to use for reading the control data. In some embodiments, the period for compiling the statistics is one second.

Referring to Fig. 13, the partial flow chart 80e illustrates an alternative step 86''' that corresponds to the step 86 of Fig. 6 where disk data that is the same in both of the cache memories 22, 24 is accessed. The step 86''' represents selecting the one of the cache memories 22, 24 from which to read the disk data using a round robin technique similar to that discussed above in connection with the step 82'' of Fig. 10 for selecting which of the cache memories 22, 24 to use for reading the control data.

Note the partial flow chart 80a includes a connector A' while the partial flow chart 80b includes a connector A''. Similarly, the partial flow chart 80c includes a connector B', the partial flow chart 80d includes a connector B'', and the partial flow chart 80e includes a connector B'''. For the system illustrated herein, the connector A' may be coupled to any of the connectors B', B'', and B'''. Similarly, the connector A'' may be coupled to any of the connectors B', B'', and B'''. Thus, the technique used to select which of the cache memories 22, 24 to use to access the control data may be somewhat independent of the technique used to access the disk data. Note also that it is possible to use one of the techniques discussed herein for accessing only the control data while using a different technique (or no technique) for accessing the disk data. Similarly, it is possible to use one of the techniques discussed herein for accessing only the disk data while using a different technique (or no technique) for accessing the control data.

Referring to Fig. 14, a diagram 150 illustrates specialized hardware 152 for providing the functionality, or at least a portion of the functionality, described above in connection with Fig.'s 9-13. The hardware 152 may be implemented using any one of a variety of technologies for designing customized hardware. The hardware 152 may be implemented using a single chip or a

plurality of chips. The hardware 152 may receive access requests for the cache via one or both of the buses 26, 28, or through some means (not shown). The hardware 152 may then process the requests in accordance with one or more of the techniques discussed above, and then read the data from one of the caches 22, 24.

Note that using the hardware 152 may reduce the requirements of keeping additional statistics because the hardware may have direct access to the queue length and other information used in connection with the techniques described herein.

While the invention has been disclosed in connection with the preferred embodiments shown and described in detail, various modifications and improvements thereon will become readily apparent to those skilled in the art. Accordingly, the spirit and scope of the present invention is to be limited only by the following claims.